Permutation-based symmetric cryptography

Guido Bertoni[1] Joan Daemen[1]
Michaël Peeters[2] Gilles Van Assche[1]

[1]STMicroelectronics

[2]NXP Semiconductors

Keccak & SHA-3 Day
Université Libre de Bruxelles
March 27, 2013

Outline

Symmetric crypto: what textbooks and intro's say

Symmetric cryptographic primitives:

- Block ciphers
- Stream ciphers
  - Synchronous
  - Self-synchronizing
- Hash functions
  - Non-keyed
  - Keyed: MAC functions

And their modes-of-use

The swiss army knife of cryptography!

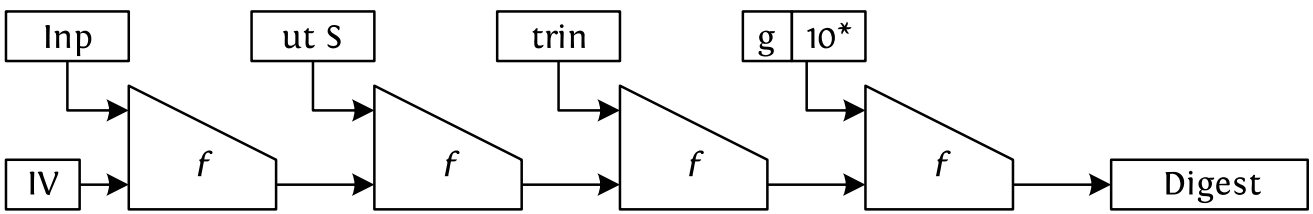
Hash functions:

A closer look at mainstream hash functions

- Attempts at direct design of hash function are rare
- Mainstream hash functions have two layers:
  - Fixed-input-length compression function
  - Iterating mode: *domain extension*

The iterating mode

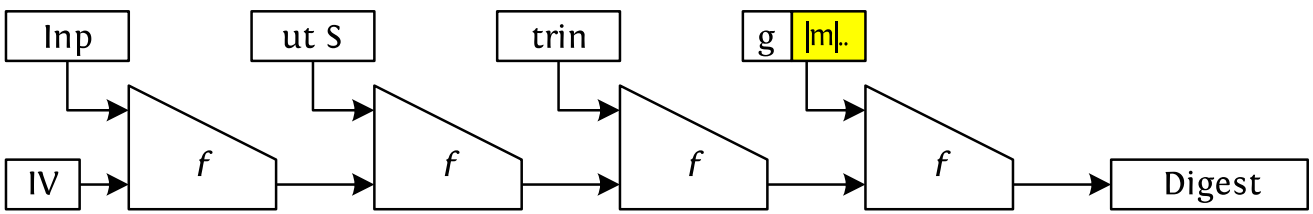
Basic Merkle-Damgård: very simple and elegant



Yes, but can we have collision-resistance preservation?

The iterating mode

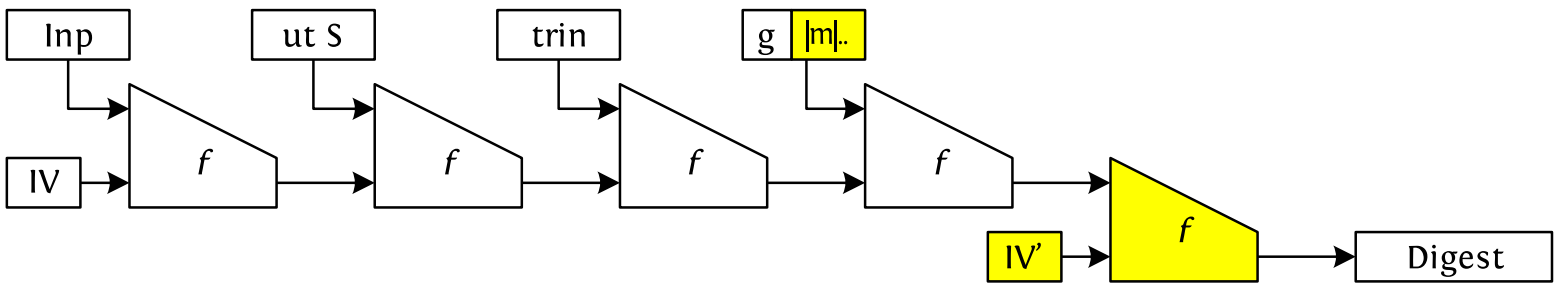
Merkle-Damgård with *strengthening*



Yes, but what about length extension attacks and the like?

The iterating mode

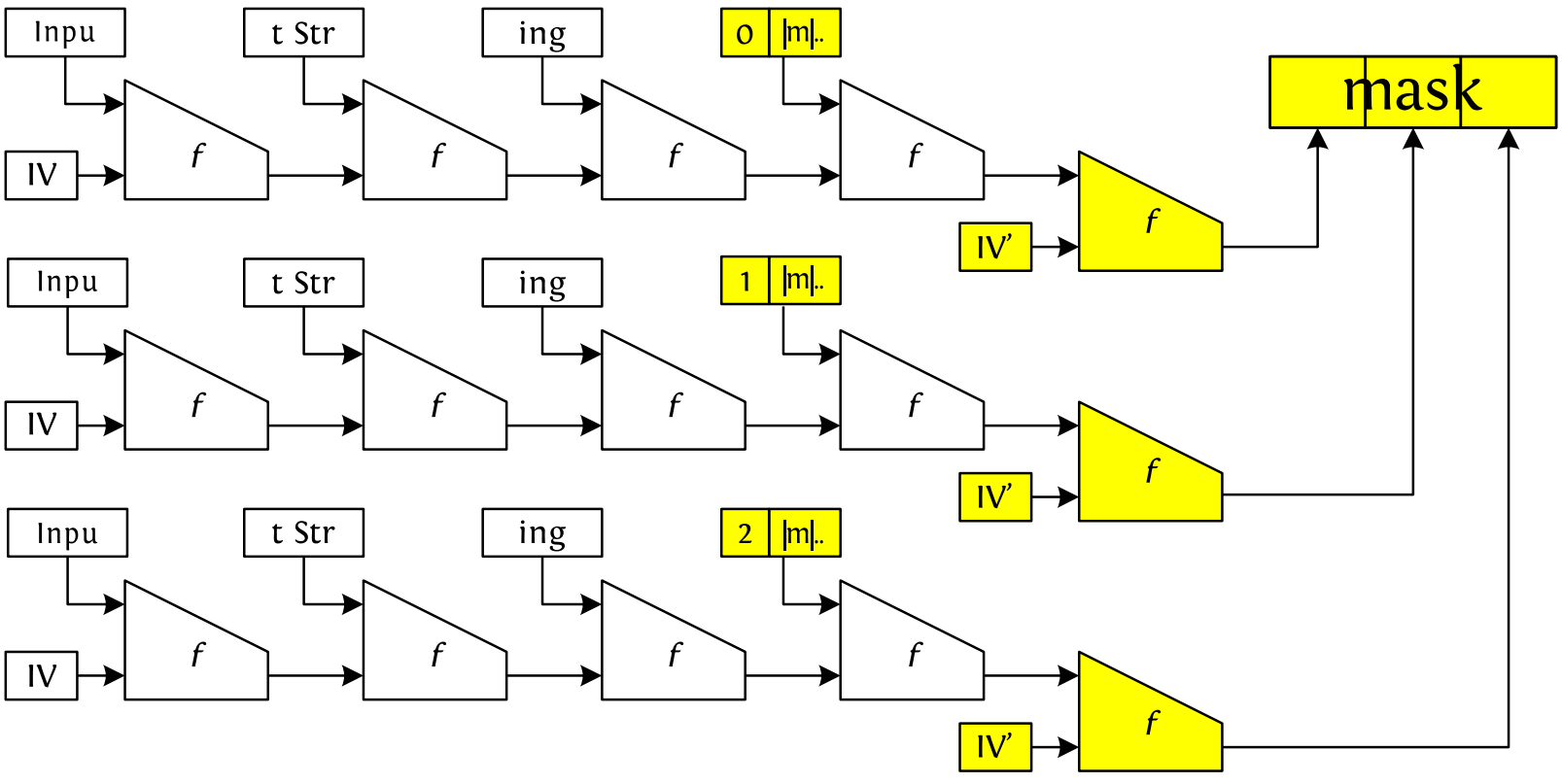
Enveloped Merkle-Damgård



Yes, but we need long output for full-domain hashing (OAEP, RSA-PSS, KEM, etc)?

The iterating mode

Mask generating function construction



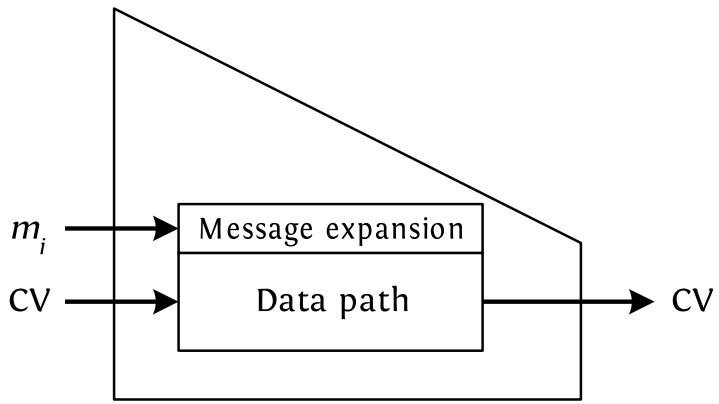
This does what we need!

The compression function

- Sound iterating mode reduces:
  - design of F with variable input and output length to
  - design of f with fixed input and output length
- Sound means there is some kind of provable security:
  - Property-preservation: F inherits the properties of f
  - Differentiability from a random oracle
    - bound for the effort to distinguish F from a random oracle
    - assuming f randomly chosen and accessible to adversary
    - Provides security assurance against generic attacks
- Gives some hints of criteria for the compression function

The compression function

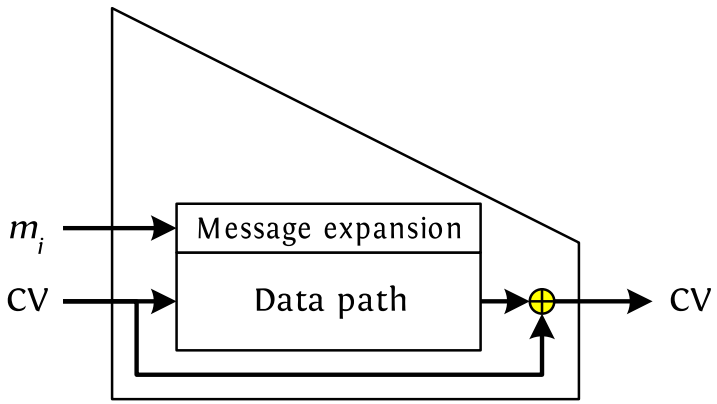
Let's put in a block cipher



Yes, but collisions are easy so collision-resistance preservation ...

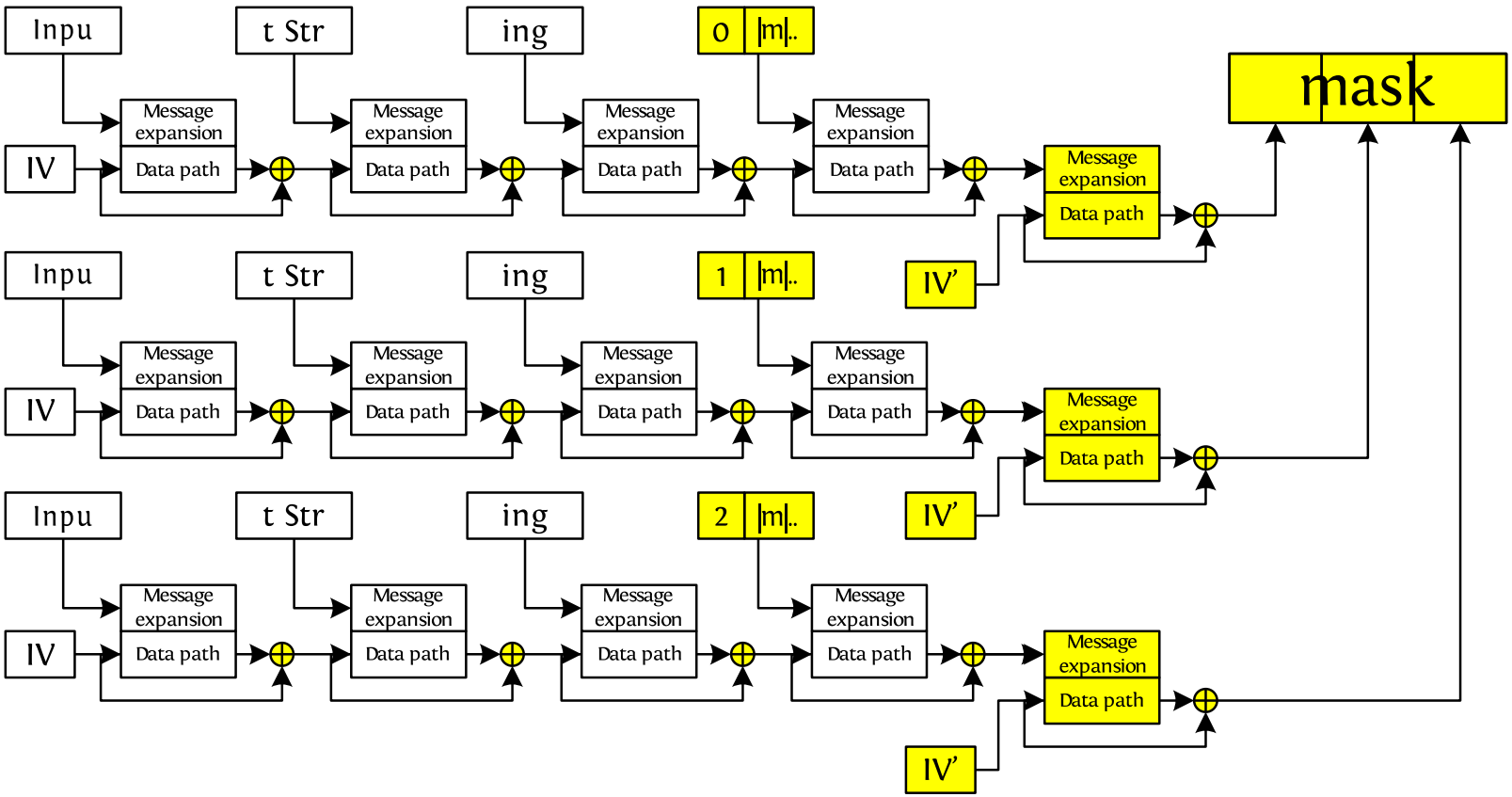
The compression function

Block cipher in Davies-Meyer mode



That's it!

The final solution



Now we just have to build a suitable block cipher ...

What block cipher are used for

- Hashing (as discussed) and its modes HMAC, MGF1, ...
- Block encryption: ECB, CBC, ...
- Stream encryption:
  - synchronous: counter mode, OFB, ...
  - self-synchronizing: CFB
- MAC computation: CBC-MAC, C-MAC, ...
- Authenticated encryption: OCB, GCM, CCM ...

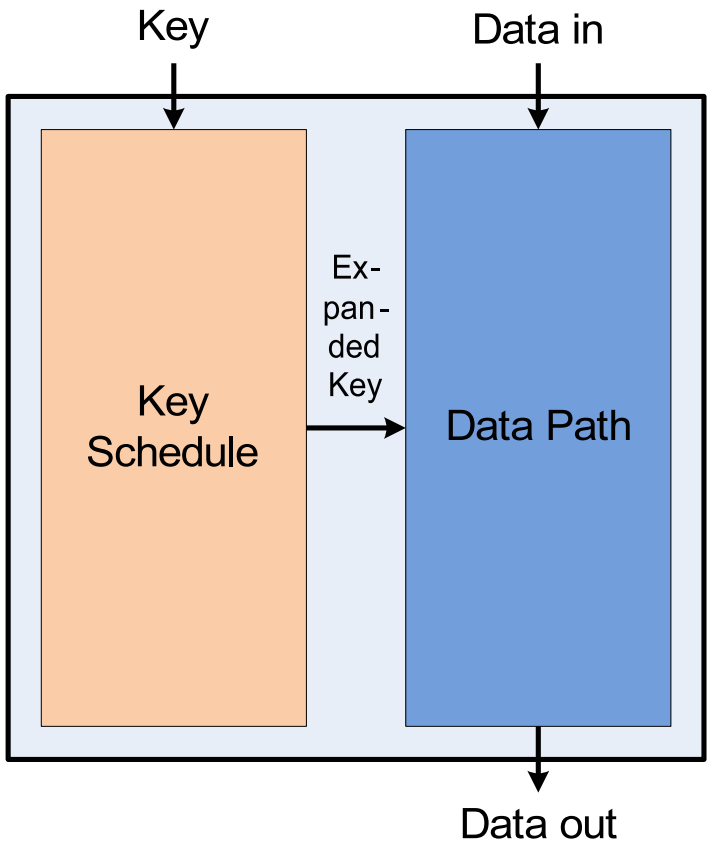
The truth about symmetric crypto today

Block ciphers:

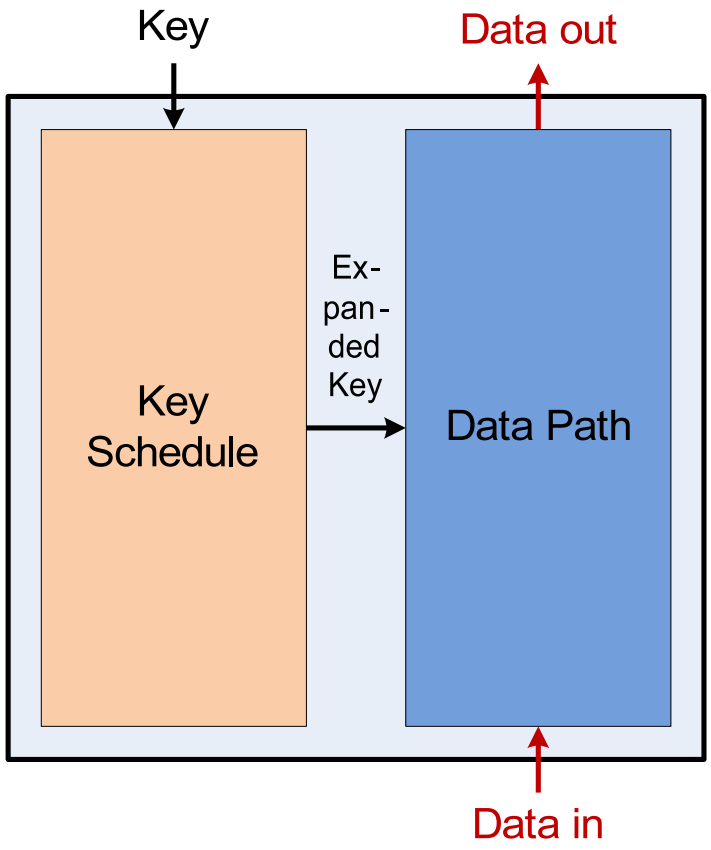
Block-cipher based hashing: time for re-factoring

- Goal: hashing mode that is sound and simple
  - with good level of security against generic attacks
  - calling a block cipher
- Remaining problem: design of a suitable block cipher
  - round function: several good approaches known
  - soundness proofs are typically in *ideal cipher* model
  - key schedule: not clear how to do design good one
- But do we really need a block cipher?

Block cipher operation



Block cipher operation: the inverse



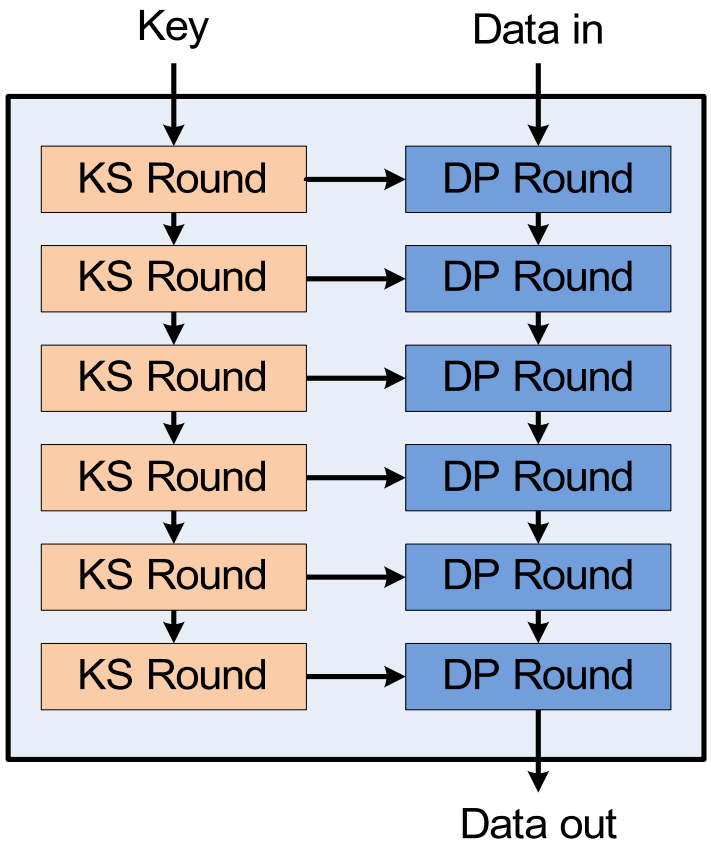
When do you need the inverse?

Indicated in red:

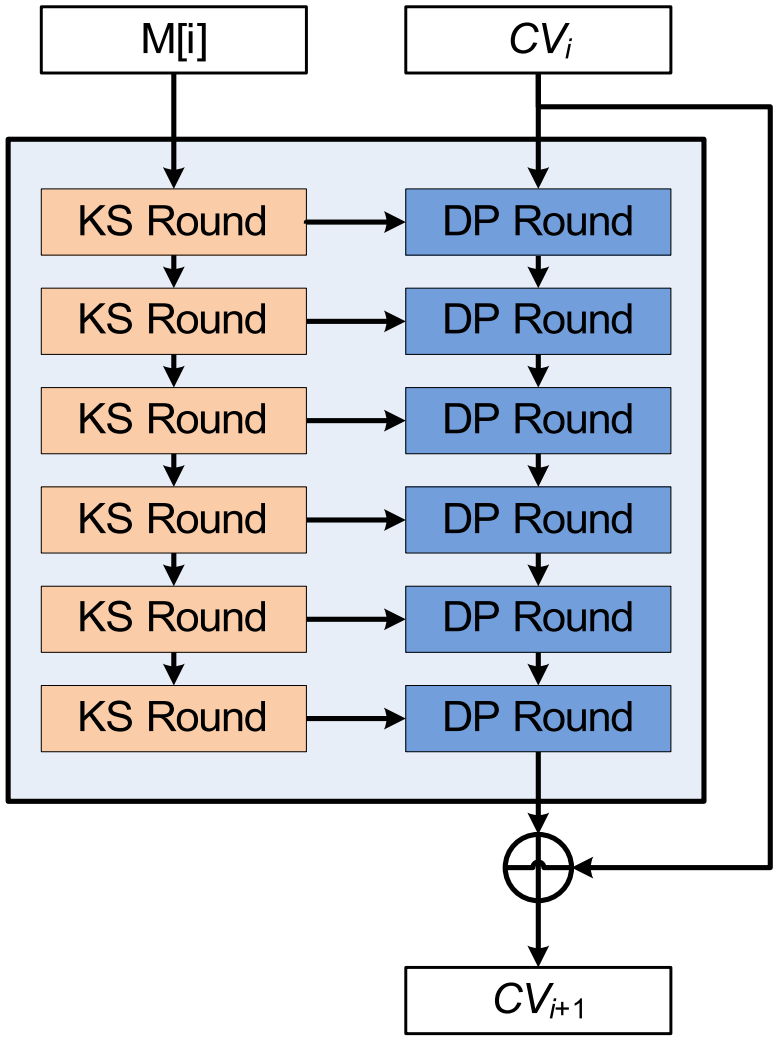
- Hashing and its modes HMAC, MGF1, ...
- Block encryption: ECB, CBC, ...
- Stream encryption:
  - synchronous: counter mode, OFB, ...
  - self-synchronizing: CFB
- MAC computation: CBC-MAC, C-MAC, ...
- Authenticated encryption: OCB, GCM, CCM ...
  - Most schemes with misuse-resistant claims

So for most uses you don't need the inverse!

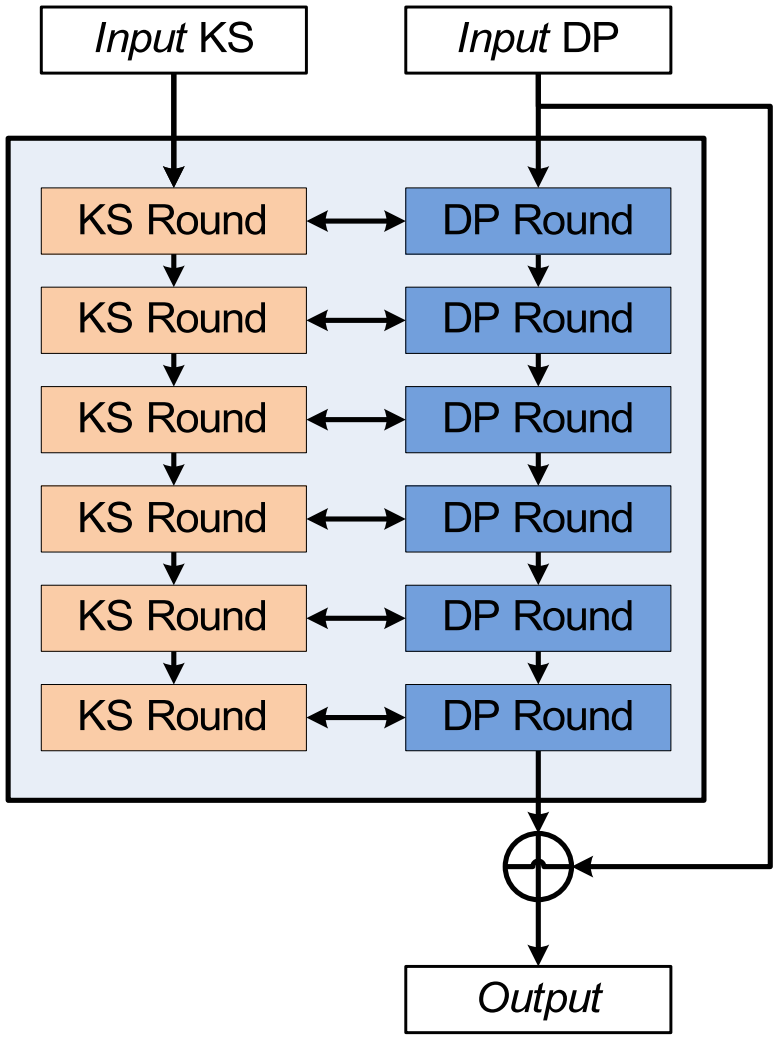
Block cipher internals



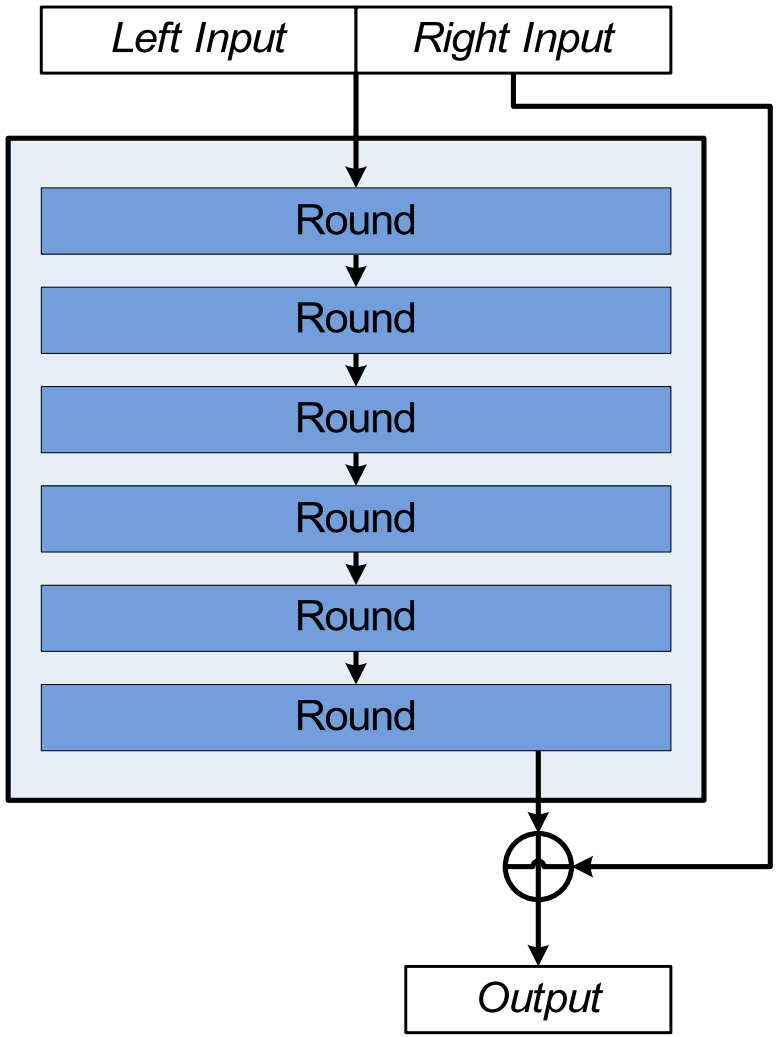
Hashing use case: Davies-Meyer compression function



Removing diffusion restriction not required in hashing



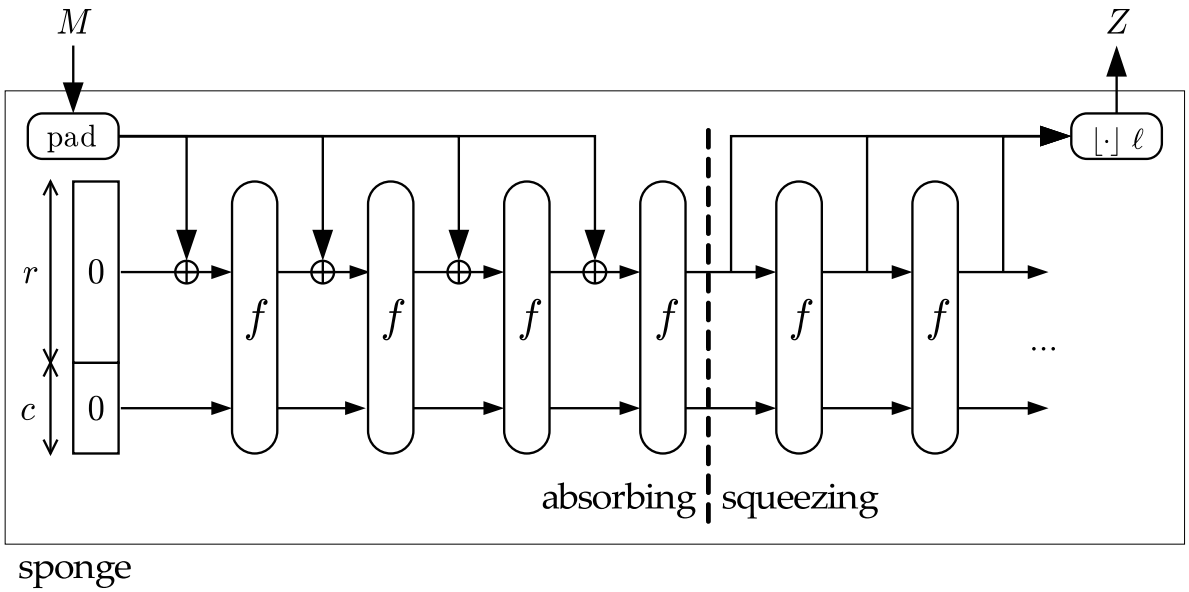
Simplifying the view: iterated permutation



Re-factoring of hashing modes, revisited

- Goal: hashing mode that is sound and simple
  - with good level of security against generic attacks
  - calling an iterated permutation
- Remaining problem: design of iterated permutation
  - round function: good approaches known
  - asymmetry: round constants
- Advantages with respect to block ciphers:
  - less barriers $\Rightarrow$ more diffusion
  - no more need for efficient inverse
  - no more worries about key schedule

The result: the sponge construction

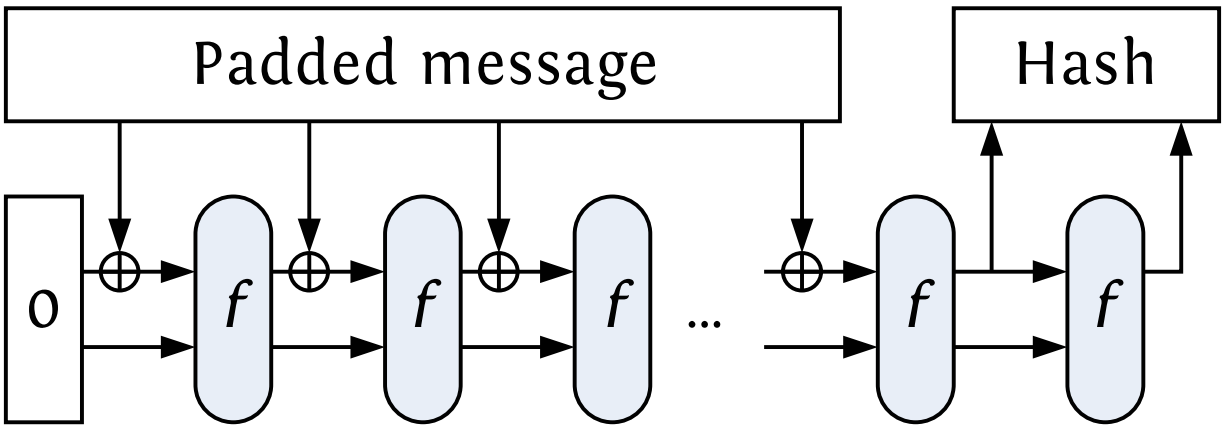


- f: a b-bit permutation with $b = r + c$
  - efficiency: processes r bits per call to f
  - security: provably resists generic attacks up to $2^{c/2}$
- Flexibility in trading rate r for capacity c or vice versa

What can we say about sponge security

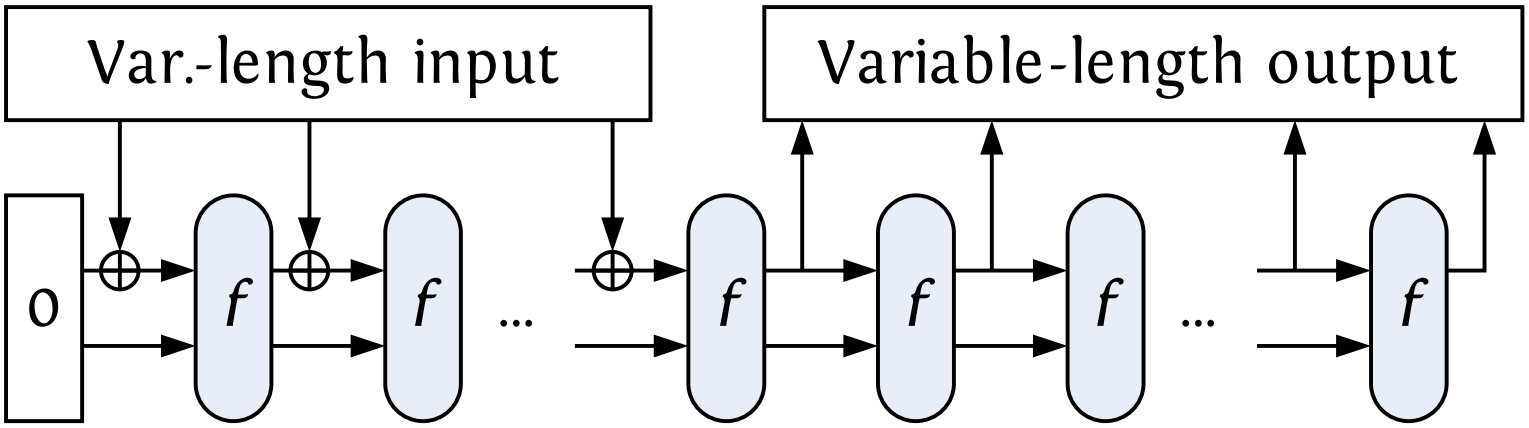
- Generic security:
  - assuming f has been chosen randomly
  - covers security against generic attacks
  - construction as sound as theoretically possible
- Security for a specific choice of f
  - security proof is infeasible
  - Hermetic Sponge Strategy
  - design with attacks in mind
  - security based on absence of attacks despite public scrutiny

Regular hashing

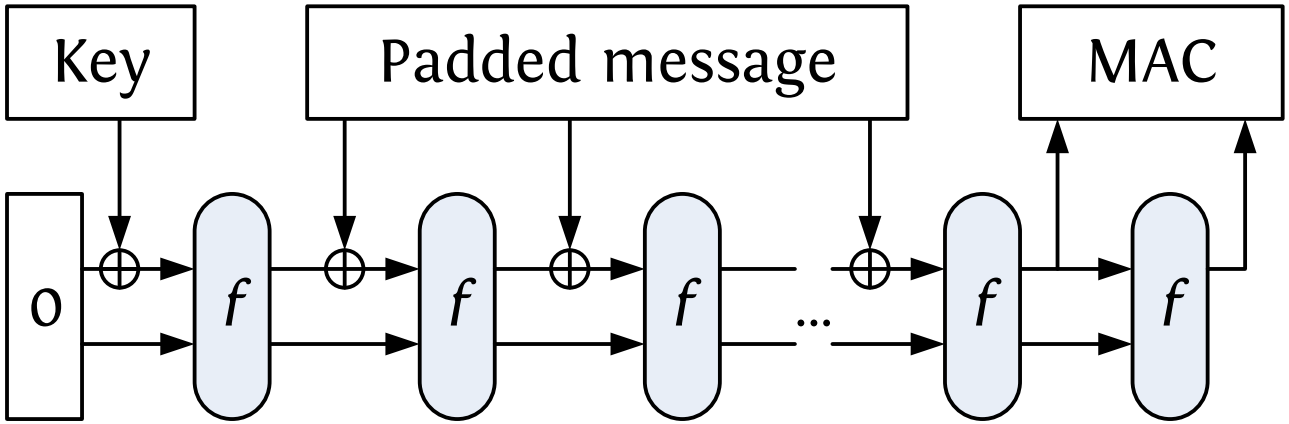


- Pre-sponge permutation-based hash functions
  - Truncated permutation as compression function: Snefru [Merkle '90], FFT-Hash [Schnorr '90], ...MD6 [Rivest et al. 2007]
  - Streaming-mode: SUBTERRANEAN, PANAMA, RADIOGATÚN, Grindahl [Knudsen, Rechberger, Thomsen, 2007], ...

Mask generating function

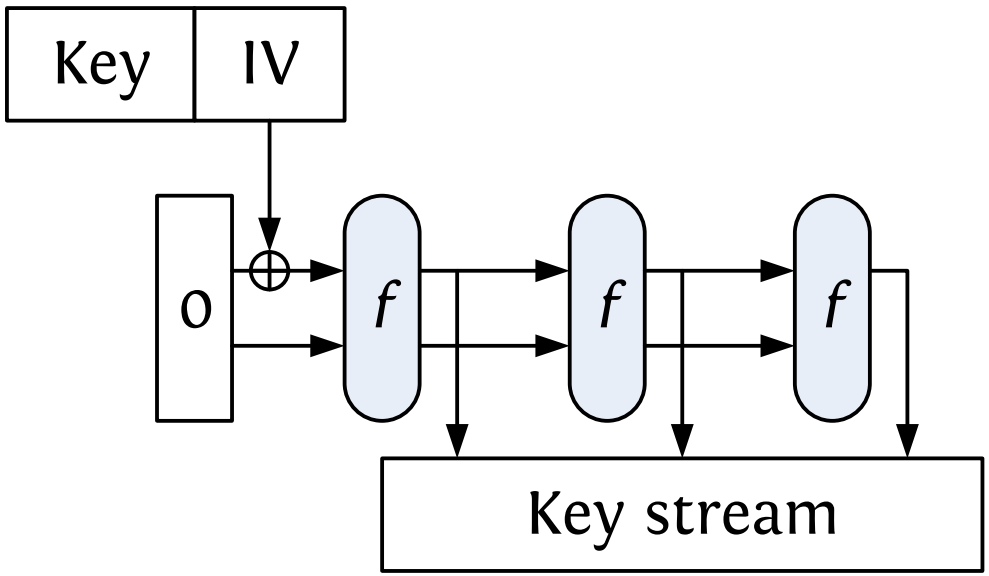


Use Sponge for MACing



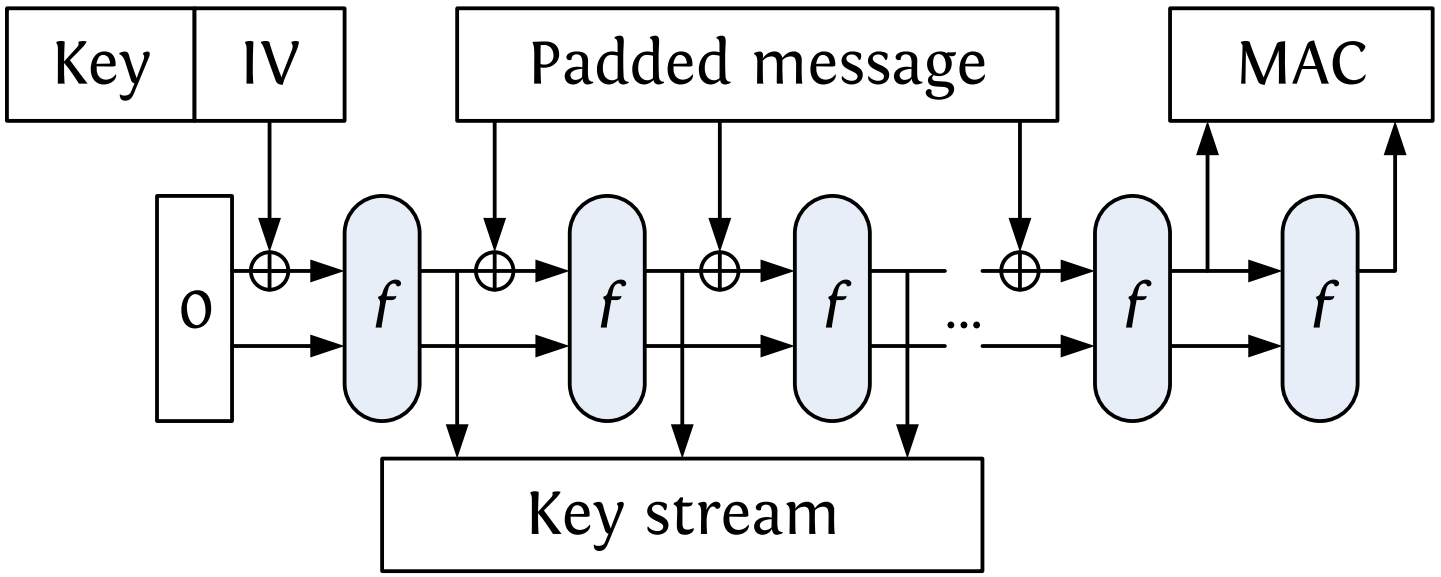
- Pre-sponge (partially) permutation-based MAC function: Pelican-MAC [Daemen, Rijmen 2005]

Use Sponge for (stream) encryption



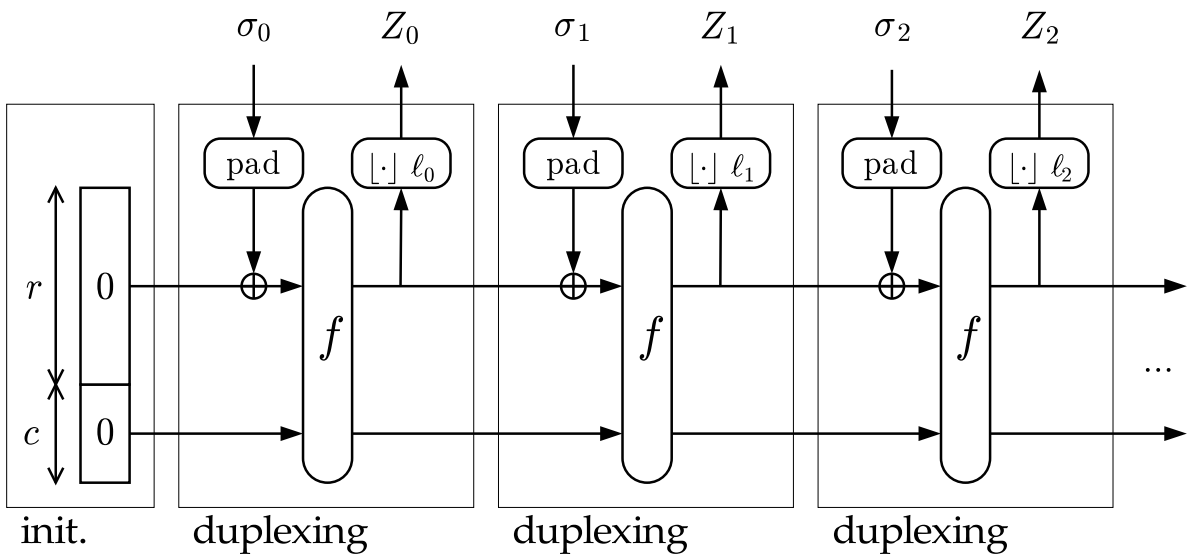
- Similar to block cipher modes:
  - Long keystream per IV: like OFB
  - Short keystream per IV: like counter mode
- Independent permutation-based stream ciphers: Salsa and ChaCha [Bernstein 2005]

Single pass authenticated encryption



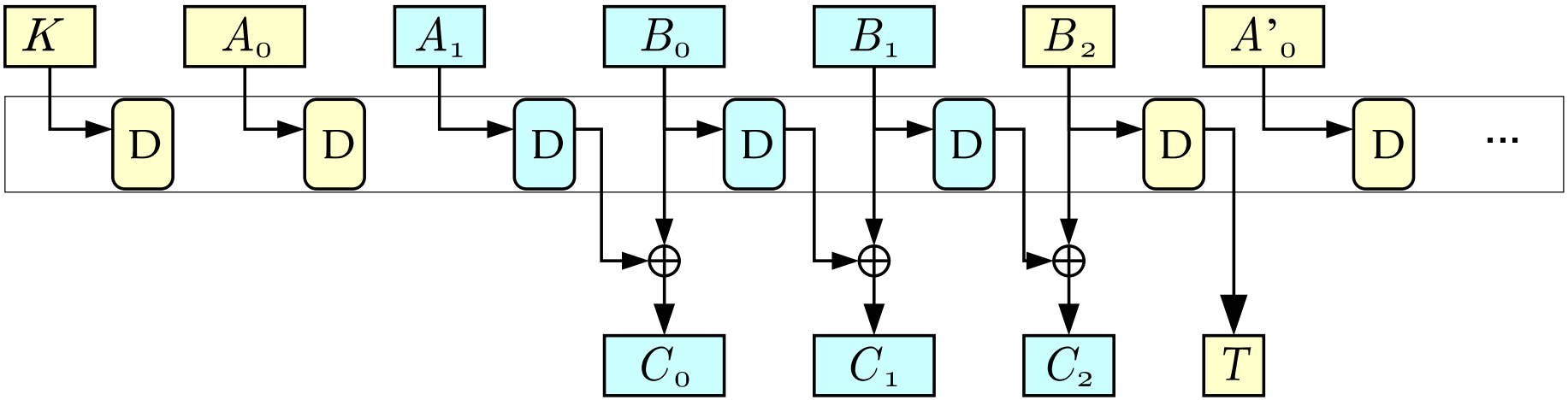
- But this is no longer the sponge ...

The duplex construction



- Generic security provably equivalent to that of sponge
- Applications: authenticated encryption, reseedable pseudorandom generator ...

SpongeWrap authenticated encryption



- Single-pass authenticated encryption
- Processes up to r bits per call to f
- Functionally similar to (P)helix [Lucks, Muller, Schneier, Whiting, 2004]

Features of iterated permutations

- Iterated permutations are
  - the natural choice for hashing modes
  - versatile cryptographic primitives
  - more flexible in modes than block ciphers
- In keyed modes efficiency can be boosted:
  - Security beyond $2^{c/2}$ in keyed modes
  - Dedicated variants for higher efficiency:
  - donkeySponge for MAC
  - monkeyDuplex for (authenticated) encryption

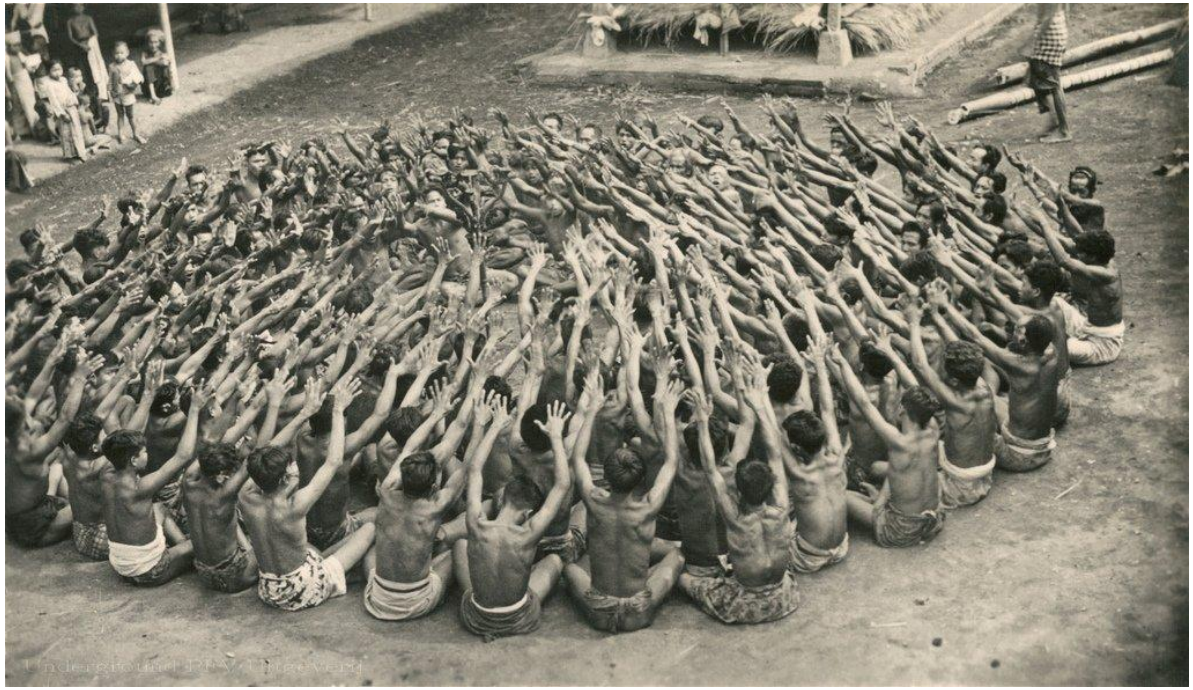
What textbooks and intro's should say from now on:-)

Symmetric cryptographic primitives:

- Permutations
- Block ciphers
- Stream ciphers
- Hash functions
  - Non-keyed
  - Keyed: MAC functions

And their modes-of-use

Questions?



http://sponge.noekeon.org/
http://keccak.noekeon.org/